sensors

Article

Accurate and Robust Monocular SLAM with Omnidirectional Cameras

Shuoyuan Liu, Peng Guo *, Lihui Feng * and Aiying Yang

The Key Laboratory of Photonics Information Technology, Ministry of Industry and Information Technology, School of Optics and Photonics, Beijing Institute of Technology, Beijing 100086, China; lsyzge405@163.com (S.L.); yangaiying@bit.edu.cn (A.Y.)

* Correspondence: guopeng0304@bit.edu.cn (P.G.); lihui.feng@bit.edu.cn (L.F.)

Received: 15 July 2019; Accepted: 14 October 2019; Published: 16 October 2019

Abstract: Simultaneous localization and mapping (SLAM) are fundamental elements for many emerging technologies, such as autonomous driving and augmented reality. For this paper, to get more information, we developed an improved monocular visual SLAM system by using omnidirectional cameras. Our method extends the ORB-SLAM framework with the enhanced unified camera model as a projection function, which can be applied to catadioptric systems and wide-angle fisheye cameras with 195 degrees field-of-view. The proposed system can use the full area of the images even with strong distortion. For omnidirectional cameras, a map initialization method is proposed. We analytically derive the Jacobian matrices of the reprojection errors with respect to the camera pose and 3D position of points. The proposed SLAM has been extensively tested in real-world datasets. The results show positioning error is less than 0.1% in a small indoor environment and is less than 1.5% in a large environment. The results demonstrate that our method is real-time, and increases its accuracy and robustness over the normal systems based on the pinhole model.

Keywords: simultaneous localization and mapping; visual SLAM; map initialization; fisheye cameras; omnidirectional cameras

1. Introduction

In order to complete various tasks, the robot needs to know the location of its environment. The most common method to localize a robot is to process the sensor information to calculate incremental motion. To achieve drift-free localization, the robot needs a map where the localization is known. To solve these problems, visual simultaneous localization and mapping (SLAM), where the main sensor is a camera, has been actively researched in recent years [1–5]. Among different sensor modalities, a monocular camera is low-cost and easy to maintain. The feature distribution of the environments observed by the camera is a key factor in the performance of Visual SLAM. A major limiting factor for the feature distribution is the field-of-view (FoV) of the camera. Especially in sparse-feature or partially non-feature environments, wide FoV cameras can get higher accuracy and better robustness than normal cameras.

However, traditional Visual SLAM can only use the part area of the wide FoV images, because these approaches are designed for the pinhole camera model which projects measurements of 3D points onto a 2D image plane. In practice, even when distortion model is added, the pinhole camera model demonstrates suboptimal performance for a FoV greater than 120°. To reduce the FoV of largely distorted image, the input images are usually cropped in traditional Visual SLAM.

To utilize the full area of the wide FoV images, it is necessary to use a new camera model that is well applied to catadioptric systems and wide-angle fisheye cameras. There are several instances of research [6–10] on camera models for large FoV cameras. Considering the accuracy and computation cost of camera models, we chose the enhanced unified camera model (EUCM) [7] as a projection function.

Since the projection function is changed, a new map initialization method should be proposed. Monocular SLAM needs a process to create an initial map because depth cannot be recovered from a single image. For planar scenes, by using the method of Faugeras et al. [11], the relative camera pose is recovered from a homography matrix. On the other hand, for non-planar scenes, the relative camera pose is recovered from a fundamental matrix which can be computed with the eight-point algorithm [12]. Because these methods are only suitable for the pinhole camera model, we propose a new initialization method based on them for the EUCM.

We improve ORB-SLAM [1,2] to make full use of the fisheye images (see Figure 1), so our method can directly use all the information of the input images.

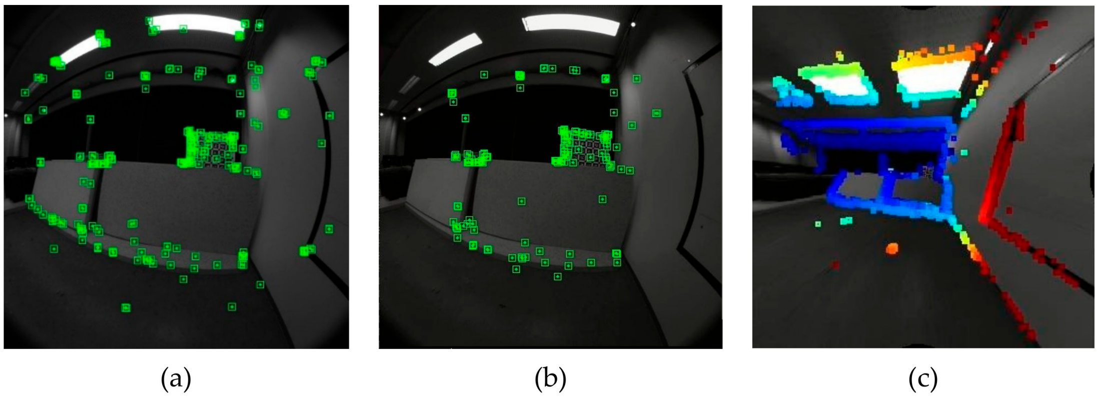


Figure 1. The selected points in a same scene: (**a**) The improved ORB-simultaneous localization and mapping (SLAM) with omnidirectional camera; (**b**) the normal ORB-SLAM; (**c**) Direct Sparse Odometry.

In the experiments, we evaluated the accuracy and run-time of our approach on a benchmark [13] whose image sequences are captured with a wide FoV fisheye lens. We compared our approach to the normal systems based on the pinhole model and demonstrated that our method outperforms the normal methods on benchmark datasets.

The rest of this paper is structured as follows: In Section 2, we first review the related systems for the normal and omnidirectional cameras. In Sections 3 and 4, we introduce notation and the enhanced unified camera model. In Section 5, we provide an overview of our system, and we analytically derive the Jacobian matrices. In Section 6, we propose a map initialization method and modify a normal relocalization algorithm for fisheye cameras. In Section 7, we quantitatively evaluate the results of our system on public datasets and compare it with the normal systems based on the pinhole model; then we discuss and interpret the results. We open source in https://github.com/lsyads/fisheye-ORB-SLAM.

2. Related Works

Over the last decades, many visual SLAM and visual odometry systems have been proposed, such as ORB-SLAM [1,2], LSD-SLAM [4], and direct sparse odometry (DSO) [3]. The three recent examples are the state-of-the-art systems for the normal cameras and points features. ORB-SLAM is an indirect and keyframe-based visual SLAM algorithm based on graph optimization. LSD-SLAM is the first direct visual SLAM method with monocular cameras, which tracks the camera motion, produces a semi-dense map and performs pose graph optimization. DSO is a direct sparse visual odometry algorithm, which combines a fully direct probabilistic model with joint optimization of all model parameters. In addition to points features, Visual SLAM for point-line [14] or point-plane [15] features has been studied for many years. Next, we will discuss the related work on omnidirectional odometry and SLAM.

By using scale-invariant feature transform (SIFT) features and the extended Kalman filter (EKF), several works [16,17] have been proposed to estimate camera localizations for omnidirectional cameras. However, these approaches lack efficient loop closing and relocalization technique, and they are only capable of mapping small work-spaces due to computational limitations.

By using a direct method, D. Caruso et al. [18] proposed a real-time, direct monocular SLAM method based on LSD-SLAM by incorporating the unified camera model [6]. Similarly, H. Matsuki et al. [19] extended DSO for omnidirectional cameras by using the unified camera model as a projection function. This system was the first fisheye-based direct visual odometry which runs in real time. L. Heng et al. [20] presented a semi-direct visual odometry for a fisheye-stereo camera. A direct visual odometry for a fisheye-stereo camera was proposed by P. Liu et al. [21].

Indirect methods are the most popular techniques for SLAM. They proceed in two steps. First, the feature points in the images are extracted and matched. Second, the geometry and camera motion are estimated through the coordinates of corresponding points. J. Li et al. [22] presented a SLAM system based on spherical model for full-view images in indoor environments, but the system was not real-time. S. Wang et al. [23] proposed to extend ORB-SLAM framework by the unified camera model and the semi-dense depth map, whose map initialization method follows the idea from [18], but there was no evaluation on localization accuracy and robustness of the system.

In this paper, we propose an omnidirectional camera extension of ORB-SLAM by using the EUCM as a projection function. This is a real-time robust monocular visual SLAM. Since the projection function is changed, we propose a new map initialization method and modify the normal relocalization algorithm.

3. Notation

Throughout the paper, bold lower-case letters (x) represent vectors, and light lower-case letters (t) represent scalars. Matrices will be represented by bold upper-case letters (H), and functions (including images) will be represented by light upper-case letters (I). Pixel coordinates generally are denoted as $\mathbf{u} = [u, v]^T \in \mathbb{R}^2$. Point coordinates in 3D are denoted as $\mathbf{x} = [x, y, z]^T \in \mathbb{R}^3$. $[\mathbf{x}]_\times$ stands for:

$$[\mathbf{x}]_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}. \tag{1}$$

The camera orientation and position are represented by $\mathbf{R} \in SO(3)$ and $\mathbf{t} \in \mathbb{R}^3$, respectively. They transform a 3D point from the camera coordinate system to the world coordinate system. Traditionally, π stands for a camera projection function and π^{-1} is the camera unprojection function.

4. Camera Models

4.1. Pinhole Model

The pinhole model is the most common camera model. Image points u are computed by projecting measurements of 3D points x onto a 2D image plane. The projection function of the pinhole model is given as

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix} \begin{bmatrix} x/z \\ y/z \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{2}$$

where f_x, f_y are the focal lengths, and c_x, c_y are the principal points. The projection function is linear in homogeneous coordinates, so it is the simplest model.

4.2. Extended Unified Camera Model

We use the enhanced unified camera model (EUCM) based on the so-called unified camera model [6] for a wide FoV fisheye camera. The projection model does not require additional mapping to model distortions, and it takes just two projection parameters more than a simple pinhole model to represent radial distortions (only one parameter more than the unified model). The unprojection function can be expressed in explicit closed-form.

As shown in Figure 2, a 3D point x in camera coordinates is first mapped to x_p by projecting it onto a second-order projection surface P, then the point x_p is mapped to q by projecting it onto the plane M of $z = 1$. The coordinates of the point q are computed as follows:

$$\begin{aligned} \mathbf{q} &= \begin{bmatrix} x/[\alpha\rho+(1-\alpha)z] \\ y/[\alpha\rho+(1-\alpha)z] \\ 1 \end{bmatrix}, \\ \rho &= \sqrt{\beta(x^2+y^2)+z^2}, \end{aligned} \tag{3}$$

where $\alpha \in [0,1]$ and $\beta > 0$. The two parameters allow us to better approximate the properties of lenses despite strong distortions. For the fisheye cameras, the EUCM projects on the ellipsoid, where $\alpha \in (0.5, 1]$.

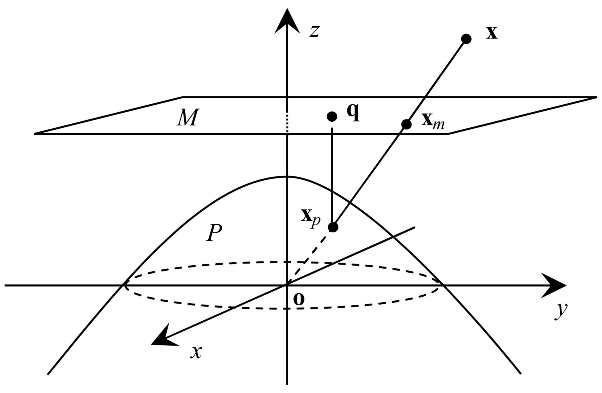


Figure 2. The enhanced unified camera model. z is the optical axis.

Finally, the image point u is computed by projecting the point q onto an image plane using the pinhole camera model (see Figure 3). The projection function of a 3D point is given by

$$\pi(\mathbf{x}) = \mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix} \begin{bmatrix} x/[\alpha\rho+(1-\alpha)z] \\ y/[\alpha\rho+(1-\alpha)z] \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}. \tag{4}$$

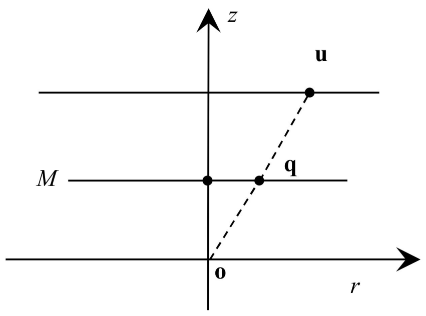


Figure 3. The pinhole camera model.

The unprojection function is defined as follows:

$$\begin{aligned}
\pi^{-1}(\mathbf{u}) &= \begin{bmatrix} m_x & m_y & m_z \end{bmatrix}^T, \\
m_x &= \frac{u - c_x}{f_x}, \\
m_y &= \frac{u - c_y}{f_y}, \\
m_z &= \frac{1 - \beta\alpha^2 r^2}{\alpha\sqrt{1 - (2\alpha - 1)\beta r^2} + (1 - \alpha)},
\end{aligned} \tag{5}$$

where $r^2 = m_x^2 + m_y^2$, and if $\alpha > 0.5, r^2 \leq 1/\beta(2\alpha - 1)$. m_z depends on m_x and m_y. Actually, $\max(m_z) = 1$.

Actually, the unprojection function (5), a ray function, is decided by the coordinates of point x_p, because points x_p and x are on the same line (see Figure 2). Plane P where point x_p is located is called *the projection surface*. The function, which solves the coordinates of point x_p from u, is the same as (5):

$$\mathbf{x}_p = \begin{bmatrix} m_x & m_y & m_z \end{bmatrix}^T. \tag{6}$$

5. System Overview

By using the EUCM as a projection function, we develop a real-time robust monocular visual SLAM system for omnidirectional cameras based on ORB-SLAM. The system makes use of ORB features [24], which are based on the FAST keypoint detector and BRIEF descriptor. Not only are they extremely fast in computation and matching, but they also have good invariance in regards of the viewpoint. In order to increase efficiency and accuracy, there are three threads in parallel: tracking, local mapping and loop closing.

5.1. Bundle Adjustment

Because Bundle Adjustment (BA) provides a powerful network of matches and good initial guesses, our system uses it to optimize the estimations of camera poses and 3D world points.

There are three kinds of BA in the system: motion-only BA, local BA, and full BA. By minimizing the reprojection error between matched 3D points x in world coordinates and keypoints u, BA optimizes the camera position t, orientation R, and points x. In motion-only BA, it only optimizes camera poses:

$$\begin{aligned}
\{\mathbf{R}, \mathbf{t}\} &= \underset{\mathbf{R},\mathbf{t}}{min} \sum_{i \in \chi} \rho(e_i^T \Sigma_i^{-1} e_i), \\
e_i &= \mathbf{u}_i - \pi(\mathbf{R}\mathbf{x}_i + \mathbf{t}),
\end{aligned} \tag{7}$$

where χ means the set of all matches, ρ is the robust Huber cost function, and $\Sigma_i = \sigma_i^2 \mathbf{I}_{2\times 2}$ is the covariance matrix associated to the scale where the keypoint was detected.

In addition to optimizing camera poses, local BA also optimizes points and minimizes the reprojection error in a collection of nearby keyframes. In full BA, all keyframes are optimized to get camera poses and points.

The Levenberg–Marquardt algorithm implementation in g2o [25] is used to solve this minimization problem. In BA, we modified the projection function to EUCM and Jacobian matrices.

5.2. Jacobian Matrices

Jacobian matrices are used in the Levenberg–Marquardt algorithm to speed up a bundle adjustment process in the SLAM system, because they are more efficient than numeric or automatic differentiation. A 3D point in world and camera coordinates are represented by $\mathbf{x} = [x, y, z]^T$ and $\mathbf{x'} = [x', y', z']^T$, respectively. The reprojection error (7) is written as e. The small changes of camera rotation and position are represented by $\delta\mathbf{R}$ and $\delta\mathbf{t}$, respectively.

The projection relation is $\mathbf{u} = \pi(\mathbf{x}')$ from (4), and (3) is reduced to $\mathbf{m} = [x'/(\alpha\rho + (1-\alpha)z'), y'/(\alpha\rho + (1-\alpha)z')]^T$.

By using the chain rule, the Jacobian matrices of the reprojection errors with respect to the camera rotation, camera position, and 3D points is computed, respectively:

$$\mathbf{J_R} = \frac{\partial e}{\partial \delta\mathbf{R}} = \frac{\partial e}{\partial \mathbf{x}'}\frac{\partial \mathbf{x}'}{\partial \delta\mathbf{R}} = \frac{\partial e}{\partial \mathbf{x}'}(-[\mathbf{x}']_\times), \tag{8}$$

$$\mathbf{J_t} = \frac{\partial e}{\partial \delta\mathbf{t}} = \frac{\partial e}{\partial \mathbf{x}'}\frac{\partial \mathbf{x}'}{\partial \delta\mathbf{t}} = \frac{\partial e}{\partial \mathbf{x}'}I = \frac{\partial e}{\partial \mathbf{x}'}, \tag{9}$$

$$\mathbf{J_x} = \frac{\partial e}{\partial \mathbf{x}} = \frac{\partial e}{\partial \mathbf{x}'}\frac{\partial \mathbf{x}'}{\partial \mathbf{x}} = \frac{\partial e}{\partial \mathbf{x}'}\mathbf{R}, \tag{10}$$

where

$$\begin{aligned} \frac{\partial e}{\partial \mathbf{x}'} &= -\frac{\partial \mathbf{u}}{\partial \mathbf{x}'} = -\begin{bmatrix} f_x & 0 \\ 0 & f_y \end{bmatrix}\frac{\partial \mathbf{q}}{\partial \mathbf{x}'}, \\ \frac{\partial \mathbf{q}}{\partial \mathbf{x}'} &= \begin{bmatrix} \frac{1}{\eta} - \frac{\alpha\beta x'^2}{\eta^2\rho} & -\frac{\alpha\beta x'y'}{\eta^2\rho} & -\frac{x'(1-\alpha+(\alpha z')/\rho)}{\eta^2} \\ -\frac{\alpha\beta x'y'}{\eta^2\rho} & \frac{1}{\eta} - \frac{\alpha\beta y'^2}{\eta^2\rho} & -\frac{y'(1-\alpha+(\alpha z')/\rho)}{\eta^2} \end{bmatrix}, \\ \eta &= \alpha\rho + (1-\alpha)z'. \end{aligned} \tag{11}$$

The Jacobian matrixes for the EUCM are different from the Jacobian matrixes for the pinhole model, so it is necessary to analytically derive the Jacobian matrices.

5.3. Tracking, Local Mapping, and Loop Closing

The tracking is responsible for positioning the camera per frame and deciding when to insert a new keyframe. To get the initial camera pose and initial map points, we first perform a map initialization for omnidirectional cameras, which is explained in detail in Section 6.1. If the initialization is successful, we optimize the pose using *motion-only BA*. Then a local map is acquired. It contains the map points of nearby keyframes. Matches with the local map points are searched by reprojection, then the camera pose and the map points are optimized with all matches. Finally, the tracking thread decides if a new keyframe is inserted. When the tracking is lost, the relocalization module for omnidirectional cameras starts working, which is explained in Section 6.2.

The local mapping processes the new keyframes and executes *the local BA* to optimize camera poses and correct map point positions. New points are created by using a triangulation method which is the same as the triangulation of initialization for omnidirectional cameras.

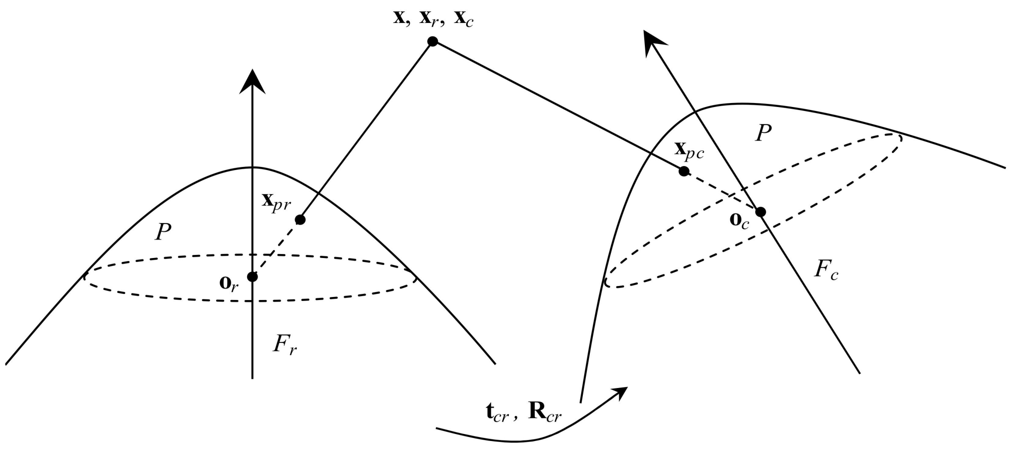
The loop closing is in charge of searching for loops with every new keyframe. For omnidirectional cameras, this thread is almost the same as the original thread except *the full BA* and projection function, which should be the EUCM instead of the pinhole model.

6. Map Initialization and Relocalization Algorithm

6.1. Map Initialization Algorithm

The goal of the map initialization is to get the relative pose and triangulate a set of initial map points from matching feature points of two frame. Two geometrical models, the homography matrix assuming a planar scene and the essential matrix assuming a non-planar scene, are computed in parallel. Then a suitable model is automatically selected, and the relative pose is recovered with the selected model.

The method first extracts ORB features in the current frame F_c and reference frame F_r. It then searches for matches $\mathbf{u}_c \leftrightarrow \mathbf{u}_r$. By (6), the coordinates of points $\mathbf{x}_{pc} = \left[m_{xc}, m_{yc}, m_{zc}\right]^T$ and $\mathbf{x}_{pr} = \left[m_{xr}, m_{yr}, m_{zr}\right]^T$ (see Figure 4) are solved from $\mathbf{u}_r$ and $\mathbf{u}_c$, respectively, so there are *new matches* $\mathbf{x}_{pc} \leftrightarrow \mathbf{x}_{pr}$. A point position in world, a current frame and the reference frame homogeneous coordinates are represented by $\mathbf{x} = [x, y, z, 1]^T$, $\mathbf{x}_c = [x_c, y_c, z_c, 1]^T$ and $\mathbf{x}_r = [x_r, y_r, z_r, 1]^T$, respectively.



$\mathbf{x}$, $\mathbf{x}_r$, $\mathbf{x}_c$
$\mathbf{x}_{pr}$
P
$\mathbf{o}_r$
F_r
$\mathbf{x}_{pc}$
P
$\mathbf{o}_c$
F_c
$\mathbf{t}_{cr}$, $\mathbf{R}_{cr}$

Figure 4. The points $\mathbf{x}_{pc}$ and $\mathbf{x}_{pr}$ back project to rays, and the intersection of the two rays is the object point x. From the new matches $\mathbf{x}_{pc} \leftrightarrow \mathbf{x}_{pr}$, we can recover the relative pose $[\mathbf{R}_{cr}|\mathbf{t}_{cr}]$ between two frames and triangulate the point x.

As shown in Figure 4, reference frame center o_r, points x_{pr} and x_r are on the same line, and current frame center o_c, points x_{pc} and x_c are on the same line. There are two non-zero scale factors λ_r and λ_c:

$$\begin{aligned} \lambda_r \mathbf{x}_{pr} &= \mathbf{x}_r = [\mathbf{R}_r|\mathbf{t}_r]\mathbf{x}, \\ \lambda_c \mathbf{x}_{pc} &= \mathbf{x}_c = [\mathbf{R}_c|\mathbf{t}_c]\mathbf{x}, \end{aligned} \tag{12}$$

where $[\mathbf{R}_r|\mathbf{t}_r]$ and $[\mathbf{R}_c|\mathbf{t}_c]$ are 3×4 matrices which represent camera poses.

6.1.1. The Homography Matrix

The homography matrix is suitable for a planar scene where points x are located. The plane can be set to $z = 0$, so $\mathbf{x} = [x, y, 0, 1]^T$. By using (12), we get

$$\begin{aligned} \lambda_r \mathbf{x}_{pr} &= [\mathbf{R}_r|\mathbf{t}_r][x, y, 0, 1]^T = \mathbf{H}_r[x, y, 1]^T, \\ \lambda_c \mathbf{x}_{pc} &= [\mathbf{R}_c|\mathbf{t}_c][x, y, 0, 1]^T = \mathbf{H}_c[x, y, 1]^T, \end{aligned} \tag{13}$$

where $\mathbf{H}_r$ and $\mathbf{H}_c$ are 3×3 matrices.

From (13), we get

$$\lambda_c \mathbf{x}_{pc} = \lambda_r \mathbf{H}_c \mathbf{H}_r^{-1} \mathbf{x}_{pr} = \lambda_r \mathbf{H}_{cr} \mathbf{x}_{pr}, \tag{14}$$

where $\mathbf{H}_{cr}$ is the homography matrix that can recover the relative pose between two frames.

The scale factors in (14) is eliminated by the cross product:

$$\mathbf{x}_{pc} \times (\mathbf{H}_{cr} \mathbf{x}_{pr}) = 0. \tag{15}$$

There is no scale factors λ_r and λ_c in (15), so all the *new matches points* $\mathbf{x}_{pc} \leftrightarrow \mathbf{x}_{pr}$ of the two frames satisfy this equation for the same $\mathbf{H}_{cr}$. This form (15) will derive a simple linear solution to $\mathbf{H}_{cr}$, and it is solved by the normalized direct linear transformation (DLT) as explained in [12] inside a random sample consensus (RANSAC) scheme.

6.1.2. The Essential Matrix

The essential matrix is suitable for a non-planar scene where points x are located. As shown in Figure 4, point $\mathbf{x}_r$ can be transformed into $\mathbf{x}_c$ by:

$$\mathbf{x}_c = [\mathbf{R}_{cr}|\mathbf{t}_{cr}]\mathbf{x}_r, \tag{16}$$

where $[\mathbf{R}_{cr}|\mathbf{t}_{cr}]$ stands for the relative pose between current frame and reference frame.

From (16), we get the definition equation for the essential matrix $\mathbf{E}_{cr}$ in [12]:

$$\mathbf{x}_c^T\mathbf{E}_{cr}\mathbf{x}_r = 0, \tag{17}$$

where $\mathbf{E}_{cr} = [\mathbf{t}_{cr}]_\times\mathbf{R}_{cr}$.

By substituting (12) into (17):

$$\left(\lambda_c\mathbf{x}_{pc}\right)^T\mathbf{E}_{cr}\left(\lambda_r\mathbf{x}_{pr}\right) = 0. \tag{18}$$

Because $\lambda_r \neq 0$ and $\lambda_c \neq 0$, we get

$$\mathbf{x}_{pc}^T\mathbf{E}_{cr}\mathbf{x}_{pr} = 0. \tag{19}$$

There are no scale factors λ_r and λ_c in (19), so all *new matches points* $\mathbf{x}_{pc} \leftrightarrow \mathbf{x}_{pr}$ of the two frames satisfy this equation for the same $\mathbf{E}_{cr}$. The essential matrix $\mathbf{E}_{cr}$ can be solved from (19) by 8-point algorithms [12] inside a RANSAC scheme.

6.1.3. Matrix Selection

The homography matrix $\mathbf{H}_{cr}$ and the essential matrix $\mathbf{E}_{cr}$ are computed in parallel threads. Then a suitable matrix is automatically selected by a robust heuristic method [2], which chooses a matrix in $\mathbf{H}_{cr}$ and $\mathbf{E}_{cr}$ with smaller symmetric transfer errors [12].

6.1.4. Motion Recovery and Triangulation Method

We make the motion recovery from the selected matrix, then get the relative pose $[\mathbf{R}_{cr}|\mathbf{t}_{cr}]$ between two frames. In the case of the homography matrix, we retrieve 8 motion hypotheses using the method of Faugeras et al. [11]. In the case of the essential matrix, we retrieve 4 motion hypotheses using the singular value decomposition method explained in [12].

We triangulate these hypotheses and select a solution with lower reprojection error. The triangulation method, which is based on the linear triangulation method [12], will be introduced below.

From (12), we can get on *new matches points* $\mathbf{x}_{pc} \leftrightarrow \mathbf{x}_{pr}$

$$\begin{aligned} \mathbf{x}_{pr} \times ([\mathbf{R}_r|\mathbf{t}_r]\mathbf{x}) &= 0, \\ \mathbf{x}_{pc} \times ([\mathbf{R}_c|\mathbf{t}_c]\mathbf{x}) &= 0. \end{aligned} \tag{20}$$

A cross product of (20) gives three equations for each image point, of which two are linearly independent. Equation (20) can be written as four independent equations:

$$\begin{bmatrix} m_{xr}\left(\mathbf{P}_r^3\right)^T - m_{zr}\left(\mathbf{P}_r^1\right)^T \\ m_{yr}\left(\mathbf{P}_r^3\right)^T - m_{zr}\left(\mathbf{P}_r^2\right)^T \\ m_{xc}\left(\mathbf{P}_c^3\right)^T - m_{zc}\left(\mathbf{P}_c^1\right)^T \\ m_{yc}\left(\mathbf{P}_c^3\right)^T - m_{zc}\left(\mathbf{P}_c^2\right)^T \end{bmatrix}\mathbf{x} = 0, \tag{21}$$

where $\mathbf{P} = [\mathbf{R}|\mathbf{t}]$ and $\mathbf{P}^i$ are the rows of P. Equation (21) is solved by using the singular value decomposition, then the world coordinates of x are obtained.

6.2. Relocalization Algorithm

The relocalization model starts working when the tracking is lost. For omnidirectional cameras, the normal relocalization model based on ORB-SLAM is modified. The relocalization algorithm performs first an ORB matching with each candidate keyframes. The method then performs RANSAC iterations of the EPnP algorithm [26] to find a camera pose supported by enough inliers.

Because the EPnP algorithm is only suitable for the pinhole model, and the image surface must be plane, we modified the relocalization model for the EUCM. In order to meet the conditions of the EPnP algorithm, we map first an image point u to a point x_p, then we map the point x_p to a point x_m which is the intersection of the ray ox_p and the $z = 1$ plane (see Figure 2). The coordinates of x_m are computed by using (5):

$$\mathbf{x}_m = \begin{bmatrix} m_x/m_z & m_y/m_z & 1 \end{bmatrix}^T (m_z \neq 0). \tag{22}$$

The point x_m meets the conditions of the EPnP algorithm. The value of m_z is not allowed to be too small to improve the accuracy of the EPnP algorithm. In our system, we let $\min(m_z) = 0.1$, and there are very few points of $m_z < 0.1$. Basically, we use the coordinates of x_m instead of image points u in the EPnP algorithm.

7. Results and Discussion

We performed an experimental evaluation of our omnidirectional ORB-SLAM in terms of accuracy and robustness on the TUM VI benchmark [13]. The datasets provide camera images at 20Hz with a 195 degrees FoV fisheye lens. The image resolution is 512×512. We obtained the camera intrinsic parameters by using the Kalibr calibration toolbox [27] with the original checkerboard sequence. We used 3 types of the datasets: room, corridor, and magistrale. Figure 5 shows the images from the datasets. The room datasets are the sequences in the room with Motion Capture system, where ground-truth poses are available for the entire trajectory. The corridor datasets are sequences in the corridor and several offices. The magistrale datasets are sequences in the large hall. The ground-truth poses of the above two datasets are available for the start and end segments in the same room with Motion Capture system.

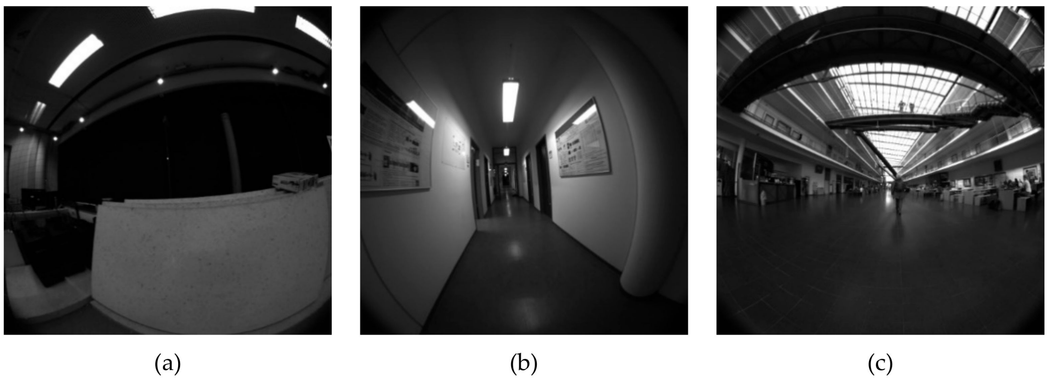


Figure 5. Sample images in the datasets: (**a**) room; (**b**) corridor; (**c**) magistrale.

We provided a quantitative comparison between our system and other systems, including the normal ORB-SLAM and DSO. Since the normal ORB-SLAM and DSO use the pinhole model as the projection function, we used the Kannala-Brandt model [8] with 8 parameters to rectify the key points in ORB-SLAM and rectify and crop the image in DSO. We evaluated both algorithms in an Intel Core i5-8300H notebook computer with 16GB RAM.

7.1. Accuracy and Robustness Comparison

We measured the absolute translational root mean square error (RMSE) between the estimated and the ground-truth position. The estimated position is computed on all keyframes and is aligned with the ground-truth trajectory data. We get the scale factor by least square method where we minimize the difference between the vector length of the estimated position and the truth position. To facilitate a fair comparison: In the room datasets, the alignment is performed for all keyframes; in the corridor and magistrale datasets, the alignment is only performed on the start segments; and we disable loop detection in the corridor and magistrale datasets except the room datasets which are used to demonstrate the loop detection for fisheye cameras. We have run each sequence 5 times in these three systems and listed median results of each time, to account for the randomness of the multithreading system. The results shown in Table 1 demonstrate our system outperforms the other systems in accuracy and robustness. The accuracy of our system is typically below 5 cm in small indoor rooms, below 15 cm in corridors and of a few meters in the large hall, because the length is longer and longer, and the image overlap between frames is smaller and smaller. The drift computed by percentage is below 0.1% in indoor rooms and corridors, and it is also below 1.5% in the large hall. In Table 1, there are many "LOST" in the normal systems, when the camera rotates and moves too fast, and the illumination variation is big.

Some representative visual results and estimated trajectories are shown in Figures 6 and 7. Figure 6a shows a room map whose points are almost active due to wide FoV of fisheye camera; Figure 6b is a map of the same room and a corridor, where datasets start and end in the same room. As shown in Figure 7, the error of all systems is very small in the start segment, but our system suffers less drift than other systems in the end segment because constraints between keyframes are stronger in our system. Due to the wider FOV and the well-maintained scale, our system performs better than the normal ORB-SLAM and DSO.

A major advantage of fisheye cameras is that more features can be observed in an input image. Figure 1 shows our system can observe more points in a frame than the normal ORB-SLAM. The image overlap between frames is also bigger, so the constraints between keyframes is stronger in our system, as shown in the red circle of Figure 8.

Table 1. Comparison of translation root mean square error (RMSE). DSO = direct sparse odometry.

Sequences	**Keyframe Trajectory RMSE (m) ("X" means LOST)**			**Length (m)**
	DSO	**Normal ORB-SLAM**	**Omnidirectional ORB-SLAM**	
corridor1	X	X	0.1252	305
corridor2	X	X	0.1349	322
corridor3	X	X	0.1360	300
corridor4	17.6904	11.2678	0.1196	114
corridor5	X	X	0.1159	270
magistrale1	X	20.3902	11.5939	918
magistrale2	13.4714	10.0507	4.7239	561
magistrale3	X	X	7.4427	566
magistrale4	X	X	9.9546	688
magistrale5	X	X	5.2586	458
room2	0.2868	0.0623	0.0466	142
room3	0.1806	0.0497	0.0446	135

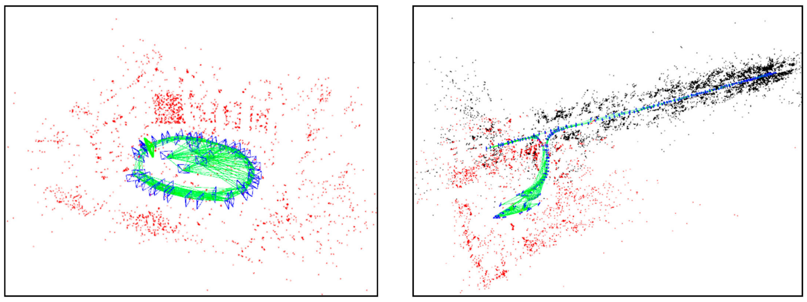
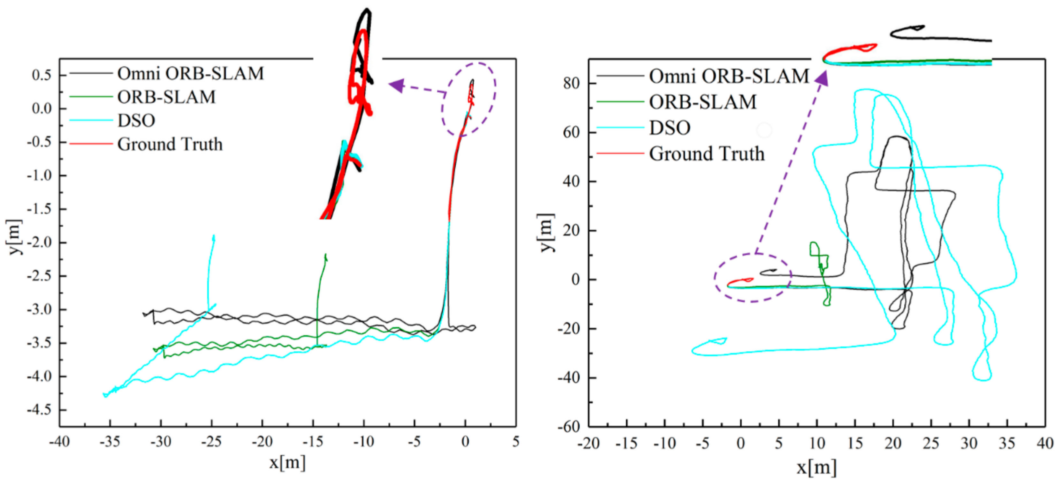

(a) (b)

Figure 6. Examples of reconstructed map of the omnidirectional ORB-SLAM. Red points are active points, black points are old points, and blue rectangles are keyframes. Dataset: (**a**) room2; (**b**) corridor4.



(a) (b)

Figure 7. Estimated trajectories. The ground truth is available for the start and end segments in the same room. Dataset: (**a**) corridor4; (**b**) magistrale2.

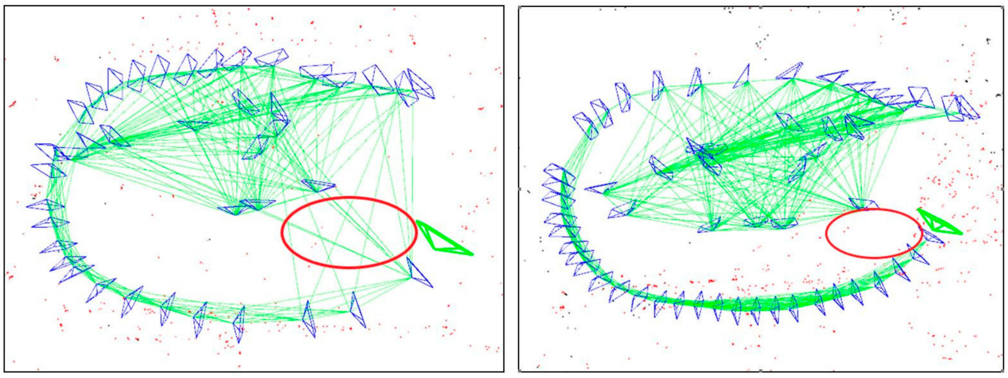

(a) (b)

Figure 8. In the red circle, the green lines represent the constraints between keyframes (blue). Our system has more lines than the normal ORB-SLAM. (**a**) The omnidirectional ORB-SLAM; (**b**) The normal ORB-SLAM. (Dataset: room2).

7.2. Timing Measurement

Table 2 shows the measured average time over the datasets for the tracking thread. These results demonstrate our system is real-time and each frame can be tracked at least 30Hz. The time cost of our system is slightly larger than the normal ORB-SLAM, because the normal ORB-SLAM needs time to rectify and crop the key points, and our system needs not rectifying and cropping the input key points but needs more time to compute on the projection function. Obviously, DSO needs more time to rectify and crop the input images and more time to compute the map points because the direct method.

Table 2. Mean Timing Results (ms).

Sequences	DSO	Normal ORB-SLAM	Omnidirectional ORB-SLAM
corridor4	137.96	24.6	30.3
magistrale2	125.17	29.1	30.8
room2	267.56	28.6	29.3

7.3. *Relocalization Experiments*

We performed two relocalization experiments on the datasets. In the experiments, we built a map with the first 40 s of the two sequences *room2* and *room3*, and performed global relocalization with every successive frame. We performed the same experiment with the normal ORB-SLAM for comparison. Table 3 shows the number of the initial map keyframes and the recall. Our system recalls better and needs less keyframes to create the initial map than the normal ORB-SLAM because more features can be observed in an input image by using fisheye cameras.

Table 3. Results for the Relocalization Experiments. KF = Kalman filter.

System	KFs of Initial Map	Recall of Relocalization (%)
	room2, 2081 frames to relocalize	
Normal ORB-SLAM	75	84.1
Omni ORB-SLAM	52	88.8
	room3, 2020 frames to relocalize	
Normal ORB-SLAM	93	61.6
Omni ORB-SLAM	61	76.1

8. Conclusions

We proposed a real-time monocular SLAM system for fisheye cameras. We first incorporated the enhanced unified camera model within the ORB-SLAM. Our system can use the full area of the images, even with strong distortion, except relocalization algorithm, which can also use more points than the normal algorithms because the points on the projection surface of EUCM are directly used. We analytically derived the Jacobian matrices to speed up the bundle adjustment process. A map initialization method was proposed for fisheye cameras. We proved *the new matches points* satisfy the homography matrix assuming a planar scene and the essential matrix assuming a non-planar scene to motion recovery and triangulation method. And we modified the relocalization algorithm for omnidirectional cameras. Then, we compared the performance of our omnidirectional ORB-SLAM and the normal systems on the public datasets, and demonstrated our method yields better performance in accuracy and robustness than the normal methods. In future work, we will extend our method with point features specially for fisheye cameras, inverse depth, and IMU.

Author Contributions: Methodology, S.L.; resources, S.L.; software, S.L.; supervision, P.G. and L.F.; validation, P.G., L.F. and A.Y.; writing—original draft preparation, S.L.; writing—review and editing, P.G., L.F. and A.Y.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 61675025.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
2. Mur-Artal, R.; Tardos, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
3. Engel, J.; Koltun, V.; Cremers, D. Direct sparse odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 611–625. [CrossRef] [PubMed]
4. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-Scale Direct Monocular SLAM. In Proceedings of the Computational Methods in Systems Biology, Manchester, UK, 17–19 November 2014.
5. Klein, G.; Murray, D. Parallel Tracking and Mapping for Small AR Workspaces. In Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13–16 November 2007; pp. 225–234.
6. Geyer, C.; Daniilidis, K. A Unifying Theory for Central Panoramic Systems and Practical Implications. In Proceedings of the European Conference on Computer Vision (ECCV), Dublin, Ireland, 26 June–1 July 2000.
7. Khomutenko, B.; Garcia, G.; Martinet, P. An enhanced unified camera model. *IEEE Robot. Autom. Lett.* **2016**, *1*, 137–144. [CrossRef]
8. Kannala, J.; Brandt, S.S. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 1335–1340. [CrossRef] [PubMed]
9. Usenko, V.; Demmel, N.; Cremers, D. The Double Sphere Camera Model. In Proceedings of the International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018.
10. Choi, K.H.; Kim, Y.; Kim, C. Analysis of Fish-Eye Lens Camera Self-Calibration. *Sensors* **2019**, *19*, 1218. [CrossRef] [PubMed]
11. Faugeras, O.; Lustman, F. Motion and structure from motion in a piecewise planar environment. *Int. J. Pattern Recognit. Artif. Intell.* **1988**, *2*, 485–508. [CrossRef]
12. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2004.
13. Schubert, D.; Goll, T.; Demmel, N.; Usenko, V.; Stuckler, J.; Cremers, D. The TUM VI Benchmark for Evaluating Visual-Inertial Odometry. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1680–1687.
14. Pumarola, A.; Vakhitov, A.; Agudo, A.; Sanfeliu, A.; Moreno-Noguer, F. PL-SLAM: Real-time monocular visual SLAM with points and lines. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 4503–4508.
15. Guo, R.; Peng, K.; Fan, W.; Zhai, Y.; Liu, Y. RGB-D SLAM Using Point-Plane Constraints for Indoor Environments. *Sensors* **2019**, *19*, 2721. [CrossRef] [PubMed]
16. Valiente, D.; Gil, A.; Payá, L.; Sebastián, J.M. Reinoso, Óscar Robust Visual Localization with Dynamic Uncertainty Management in Omnidirectional SLAM. *Appl. Sci.* **2017**, *7*, 1294. [CrossRef]
17. Valiente, D.; Gil, A.; Reinoso, Ó.; Juliá, M.; Holloway, M. Improved omnidirectional odometry for a view-based mapping approach. *Sensors* **2017**, *17*, 325. [CrossRef] [PubMed]
18. Caruso, D.; Engel, J.; Cremers, D. Large-scale direct SLAM for omnidirectional cameras. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015; pp. 141–148.
19. Matsuki, H.; Von Stumberg, L.; Usenko, V.; Stuckler, J.; Cremers, D.; Stueckler, J. Omnidirectional DSO: Direct Sparse Odometry With Fisheye Cameras. *IEEE Robot. Autom. Lett.* **2018**, *3*, 3693–3700. [CrossRef]
20. Heng, L.; Choi, B. Semi-direct visual odometry for a fisheye-stereo camera. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 4077–4084.
21. Liu, P.; Heng, L.; Sattler, T.; Geiger, A.; Pollefeys, M. Direct visual odometry for a fisheye-stereo camera. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 1746–1752.
22. Li, J.; Wang, X.; Li, S. Spherical-Model-Based SLAM on Full-View Images for Indoor Environments. *Appl. Sci.* **2018**, *8*, 2268. [CrossRef]

23. Wangl, S.; Yuel, J.; Dong, Y.; Shenl, R.; Zhang, X. Real-time Omnidirectional Visual SLAM with Semi-Dense Mapping. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 695–700.
24. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
25. Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; Burgard, W. G2o: A general framework for graph optimization. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 3607–3613.
26. Lepetit, V.; Moreno-Noguer, F.; Fua, P. EPnP: An accurate O(n) solution to the PnP problem. *Int. J. Comput. Vis.* **2009**, *81*, 155–166. [CrossRef]
27. Furgale, P.; Rehder, J.; Siegwart, R. Unified temporal and spatial calibration for multi-sensor systems. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 1280–1286.